Amendments to the Claims:

Claim amendments and status:

1. (Currently Amended) A method for ~~a~~ distributed ~~computation~~ computing, the method comprising:

defining a problem as a Cartesian grid;

obtaining a computation domain comprising one or more parallel processors; and mapping the ~~said~~ Cartesian grid to the ~~said~~ computation domain.

2. (Original) The method of claim 1 wherein said step of mapping further comprises: sub-dividing said computation domain.

3. (Original) The method of claim 2 wherein said step of sub-dividing further comprises:

defining said computation domain as a binary tree; and dividing said binary tree.

4. (Original) The method of claim 3 wherein said step of dividing further comprises:

recursively dividing said computation domain into one or more sub-domains wherein one or more processors having a shared memory remain in a common sub-domain.

5. (Original) The method of claim 1 wherein said processors are slaves and said step of mapping is performed by a master.

6. (Original) The method of claim 1 wherein said problem is a non-embarrassingly parallel problem.

7. (Original) The method of claim 3 further comprising: dynamically load balancing said computation domain, if necessary.

8. (Original) The method of claim 7 wherein said step of dynamically load balancing further comprises: performing a binary insertion operation into said binary tree.

9. (Currently Amended) An apparatus for distributed computing, the apparatus comprising:

a problem configured to be defined as a Cartesian grid;

a computation domain comprising one or more parallel processors configured to be obtained; and

a master configured to map said Cartesian grid to said computation domain.

10. (Original) The apparatus of claim 9 wherein said master further comprises: a divider configured to sub-divide said computation domain.

11. (Original) The apparatus of claim 10 wherein said divider further comprises: a binary tree configured to define said computation domain; and a second divider configured to divide said binary tree.

12. (Original) The apparatus of claim 11 wherein said second divider further comprises:

a recursive function configured to recursively divide said computation domain into one or more sub-domains wherein one or more processors having a shared memory remain in a common sub-domain.

13. (Original) The apparatus of claim 9 wherein said processors are slaves and said master is a computer.

14. (Original) The apparatus of claim 9 wherein said problem is a non-embarrassingly parallel problem.

15. (Original) The apparatus of claim 12 further comprising: a dynamic load balancer configured to dynamically load balancing said computation domain, if necessary.

16. (Original) The apparatus of claim 15 wherein said dynamic load balancer further comprises: a binary inserter configured to perform a binary insertion operation on said binary tree.

17. (Original) A computer program product comprising:

a computer usable medium having computer readable program code embodied

therein configured to distribute a computation, said computer program product comprising:

computer readable code configured to cause a computer to define a problem as a

Cartesian grid;

computer readable code configured to cause a computer to obtain a computation domain comprising one or more parallel processors;

computer readable code configured to cause a computer to map said Cartesian grid to said computation domain.

18. (Original) The computer program product of claim 18 wherein said step of mapping further comprises:

computer readable code configured to cause a computer to sub-divide said computation domain.

19. (Original) The computer program product of claim 17 wherein said computer readable code configured to cause a computer to sub-divide further comprises:

computer readable code configured to cause a computer to define said computation domain as a binary tree; and

computer readable code configured to cause a computer to divide said binary trcc.

20. (Original) The computer program product of claim 19 wherein said computer readable code configured to cause a computer to divide further comprises:

computer readable code configured to cause a computer to recursively divide said computation domain into one or more sub-domains wherein one or more processors having a shared memory remain in a common sub-domain.

21. (Original) The computer program product of claim 17 wherein said processors are slaves and said computer readable code configured to cause a computer to map is performed by a master.

22. (Original) The computer program product of claim 17 wherein said problem is a non-embarrassingly parallel problem.

23. (Original) The computer program product of claim 19 further comprising: computer readable code configured to cause a computer to dynamically load balance said computation domain, if necessary.

24. (Original) The computer program product of claim 23 wherein said computer readable code configured to cause a computer to dynamically load balance further comprises:

computer readable code configured to cause a computer to perform a binary insertion operation into said binary tree.